



ALEXANDER GALLOWAY 2018-03-21

ANTI-COMPUTER

MASHINES ANTI-COMPUTER, DIGITAL, INFORMATION MACHINE, MACHINERY, MARXISM, TECHNOLOGY, TOOL

In my graduate seminar we've recently been thinking a bit about machines. Given that our focus has been on the 19th Century, attention has been directed toward *ergodic* machines (from the root *ergon* meaning work). Ergodic machines are machines that run on heat and energy. Such machines are essentially mechanical in nature. They deal with basic physical mechanics like levers and pulleys, and questions of mass, weight, and counter-balance. Ergodic machines adhere to the laws of motion and inertia, the conservation of energy, and the laws of thermodynamics governing heat, pressure, and energy.

I've often struggled to pinpoint the difference between a tool and a machine; it's not simply a question of scale or complexity. Still, the tool and the machine constitute two different branches in the philosophy of technology. For instance Heidegger wrote about tools but had much less to say about machines. Deleuze, for his part, was obsessed with machines, leaving tools by the wayside. Overall, ergodic machines are interesting from a philosophical point of view, given how philosophy tends to privilege presence and being. Categories like energy, heat, power, change, motion, evolution, or process tend to get second billing in philosophy, if they're addressed at all. To promote them to primary billing, as Foucault did, or Whitehead, or Nietzsche, is something of a radical gesture.

Marx knew that tools were important, but it was machines that captured his attention. He devoted long sections to machines in *Capital*, vol. 1, as well as in other writings. The machine was necessary to his analysis because it played a crucial role in the increase in productivity, which in turn leads to an increase in the extraction of surplus-value. But Marx also marveled over the ergodic nature of machines. Machines were part of that vital circuit in which energy is released and translated into value. In a famous passage Marx described the way in which the "motion" and "unrest" of labor transforms itself into the static categories of being and object.

"During the labour process, the worker's labour constantly undergoes a transformation, from the form of unrest [*Unruhe*] into that of being [*Sein*], from the form of motion [*Bewegung*] into that of objectivity [*Gegenständlichkeit*]" (Marx, *Capital*, vol. 1, 296)

Machines cannot produce value on their own but are essentially large batteries for value. Nevertheless under conditions of large-

scale industrialization, machines take on an inordinately powerful role, inverting the agency of human and machine. Under the capitalist mode of production, the worker became a mere “attendant” to the machine, wrote Marx, and all labor was essentially converted into child labor.

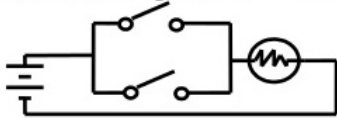


Still, ergodic machines do not account for all machines. Informatic machines, those devices dominating contemporary life, have in many ways taken over from their 19th-century counterparts. Informatic machines have physical bodies, of course, and they frequently require electricity or other forms of power to operate. However the essence of the informatic machine is not found in motion, unrest, heat, or energy. The essence of the informatic machine is found in form, not energy or presence. From the perspective of philosophy, computers are therefore quite classical, even conservative. They follow that most basic law of Western idealism, that *the formal determines the physical*.

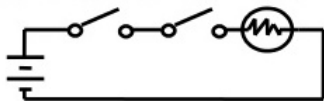
Informatic machines prevail in situations of great susceptibility to differentiation. This is why informatic machines rely so heavily on arithmetic discretization. (And why the integers are the most fundamental medium of computation.) But informatic machines will be found in any situation in which differentiation takes over. Hence weaving, one of the first human activities to be fully industrialized, was also one of the first human activities to be fully computerized. I mean computerized in a loose sense. The Jacquard loom of the early 19th Century had software imprinted on articulated cardboard ribbons. The looms used a control mechanism to execute this software on the appropriate hardware (the loom itself). Weaving's own susceptibility to differentiation is what made this process so easy. Threads are spun into discrete strands from a puffy woolen mass. Warp threads are arranged in vertical lines, facilitating discretization in one direction. The weft thread passes discretely under or over the warp threads, continuing the process of discretization. One direction is oriented perpendicular to the other, accentuating the capacity for difference. The textile grid, with its interacting orthogonality, is a perfect embodiment of digital media. (To be sure, weaving is truly contemptible from a Deleuzian perspective — practically fascist — given such an extreme form of striation and structure. Only felt, with its tangled, smooth heterogeneity, would pass the Deleuzian test.)

Differential abstraction is the key technology here. And of course there are certain special mechanisms that facilitate differential abstraction. These may be physical, like switches and logic gates. Or they may be structural, like the symbol (the number), the array or list, the tree or the graph, or the grid. Switches are absolutely crucial for informatic machines. The “And” logic gate is produced by wiring two separate switches in series. While the “Or” gate is produced by wiring two switches in parallel. (It remains to be seen what effect this might have on Kittler's famous “serial” media of 1900, and the “parallel” media that came to dominate a hundred years later. But provisionally, we might say that the gramophone and its analog ilk are “And-media,” while computers are “Or-media.”)

Switches in parallel => OR



Switches in series => AND



Still, what would it mean to suspend the computer? Or to oppose it? What are all the things that the computer can't do? If computers have been designed and optimized in certain ways, then would the inversion of such optimizations constitute something like an anti-computer?

Form wants to be free. Form wants to transcend the body, to graduate more fully into pure abstraction, to migrate into structure, into the symbolic order. In this sense, form is a gateway into exchange and equivalency. Form is *money*, or at least aspires towards the money form. Recall how, in Marx, only the “M” in the circuit of M-C-M (Money-Commodity-Money) contributes to the development of capitalism. There’s something special about that “M,” about unadulterated quantity. Pure quality does not produce the same outcome, Marx argued, because commodities “exit” the circuit once consumed (even if they reappear again, later, through the very body of the worker). Money can’t be consumed — at least you can’t eat it — it can only be spent, thereby pushing the cycle of exchange one revolution further.

When discretized, form becomes explicitly linguistic. (Non-discrete form exists, of course, but not for modern computers.) Discretized form means form as grammar, or logic, rules of composition. This is form in the realm of the semiotic, particularly the signifiers that constitute the tactile and visual body of the linguistic sign. Computational form wants to be frictionless. It wants to remove redundancy, to remove ambiguity. For every input, there shall be exactly one output. And for every output, exactly one input.

Given time, this conversation could continue indefinitely, itemizing all of the particular formal details at work in computational form. Could these details be suspended? Perhaps. Although a more pressing question today might be how to *oppose* them. For every positive affordance of the computer, one might assert an equally positive counter affordance. For every capacity, an equally potent counter capacity.

Thus given, say, an image compression format like JPEG, it becomes relatively easy to posit the “ideal” image for such a system. The ideal image would be monochromatic and motionless. The apex of digital imaging, therefore, would be found in a motionless field of color; it would be found in slow cinema. After all, androids don’t dream of electric sheep, they dream of efficient algorithms. And algorithms love simple datasets. The digital video encoder has never been happier than when given the job of encoding films by Chantal Akerman or James Benning. These are computer-friendly substrates. Sure, these films convey analog themes and anti-computational aesthetics, but then again aesthetics are always anti-computational.

As contrast, it would also be easy to generate an image at the antipode of computational form. (For maximum perversity, it ought to be generated by a computer.) This anti-image would have maximum textural complexity, and maximum kinetic complexity. It would be the pictorial form of pure entropy, a wide channel with continuous chaos. It would be video snow.

The anti-computer has yet to be invented. But traces of it are found everywhere. Even Bitcoin, that most miserable invention, relies on an anti-computational infrastructure. In order to mine coins, one must expend energy. Hence these twenty-first-century machines are yoked to a nineteenth-century mandate: burn fuel to release value. Bitcoin may run on a computer but it is anti-computational at heart. Bitcoin only works because it is grounded in an anti-computer (energy). It is thus a digital machine made subsidiary to an analog foundation, a twenty-first-century future tied to a nineteenth-century past.

The encryption algorithms at the heart of Bitcoin are anti-computational as well. Cryptography deploys form as a weapon against form. Such is the magic of encryption. Encryption is a kind of structure that makes life difficult for other competing structures. Encryption does not promote frictionlessness, on the contrary it produces full and complete friction at all levels. Not the quotidian friction of everyday life, but a radical friction frustrating all expression. What used to be a marginal activity practiced by hackers — cracking password hashes — is now the basis of an entire infrastructure. Earn a buck by cracking hashes using “brute force.” Turn your computer into an anti-computer.

taken from here

← PREVIOUS NEXT →

META

CONTACT

FORCE-INC/MILLE PLATEAUX

IMPRESSUM

DATENSCHUTZERKLÄRUNG

TAXONOMY

CATEGORIES

TAGS

AUTHORS

ALL INPUT

SOCIAL

FACEBOOK

INSTAGRAM
TWITTER